CalibRCNN: Calibrating Camera and LiDAR by Recurrent Convolutional Neural Network and Geometric Constraints

Jieying Shi[1], Ziheng Zhu[1], Jianhua Zhang[1], Ruyu Liu[1], Zhenhua Wang[1], Shengyong Chen[2] and Honghai Liu[3]

***Abstract*— In this paper, we present Calibration Recurrent Convolutional Neural Network (CalibRCNN) to infer a 6 degrees of freedom (DOF) rigid body transformation between 3D LiDAR and 2D camera. Different from the existing methods, our 3D-2D CalibRCNN not only uses the LSTM network to extract the temporal features between 3D point clouds and RGB images of consecutive frames, but also uses the geometric loss and photometric loss obtained by the interframe constraint to refine the calibration accuracy of the predicted transformation parameters. The CalibRCNN aims at inferring the correspondence between projected depth image and RGB image to learn the underlying geometry of 2D-3D calibration. Thus, the proposed calibration model achieves a good generalization ability to adapt to unknown initial calibration error ranges, and other 3D LiDAR and 2D camera pairs with different intrinsic parameters from the training dataset. Extensive experiments have demonstrated that our CalibRCNN can achieve state-of-the-art accuracy by comparison with other CNN based methods.**

I. INTRODUCTION

With the increasing development of machine vision, multi-sensor fusion data is increasingly being applied to robotic environment-aware tasks such as autonomous driving. The fusion of different sensor data relies on accurate external calibration. The most commonly used sensors are 2D camera and 3D LiDAR in the perception stage. The former can capture rich environment information including texture and color, and the latter can acquire accurate range measurements of distance with a wide angular view. Combining these sensors in SLAM system can overcome the individual sensor limitations [1], [2].

In this work, we present CalibRCNN to solve the problem of LiDAR-camera calibration online. Our method is mainly used to calibrate the calibration deviation of LiDAR and camera in real-time application environment, such as autonomous driving platform. Drawing on the successful application of deep learning in computer vision [3]–[5], we use neural networks to extract geometric and photometric features of synchronized different sensor data. Different from existing CNN-based calibration methods, we integrate LSTM to extract temporal features after the CNN module. Then, the 6-DOF rigid body transformation between 3D LiDAR and 2D camera is decoupled from the fused deep features. By considering the constraint of relative pose between 3D LiDAR and 2D camera in continuous frames, the training of the network is based on the calibration error of projected depth map of the point cloud, the photometric and geometric error obtained by the inter-frame pose transformation, and the predicted parameter error, which is also a plus over existing calibration methods. The architecture of our system is shown in Fig.1.

To evaluate the performance of the proposed method, we train the model on consecutive pairs of images and point clouds extracted from some sequences of KITTI dataset. And the model is then used to infer the transformation between LiDAR and camera in different sequences.

The main contributions of this paper include:

1) It is a end-to-end approach on LiDAR-camera calibration problem, by combining CNN and LSTM for feature extraction and feature matching.
2) We use a synthetic view and an epipolar geometry constraint to quantify the photometric and geometric errors between *successive* frames to optimize the calibration model.
3) We found that the calibration model can perform better in a wide-field environment and has a certain generalization ability.

II. RELATED WORK

In the past few years, many calibration techniques have been proposed, especially for LiDAR-camera calibration problems [6]–[16]. In general, these techniques can be divided into two groups, i.e., off-line and online. Off-line methods [6]–[10] require significant amounts of manual effort, and calibrate with specific targets. The limitation of these methods is that good parameters of the off-line calibration can not always accurately calibrate LiDAR and camera after environmental changes or vibrations like some bumps and jolts in real-time applications. Traditional online methods [11]–[13] are proposed to overcome these deficiencies. These methods can gradually converge to accurate parameters by handling continuously input images and point clouds. In recent years, several methods have emerged using deep learning for online sensor calibration, such as [14]–[16]. They mainly employ the convolutional neural network

This work was partially supported by National Key R&D Program of China (2018YFB1305200). This publication was partially funded by the National Natural Science Foundation of China (61876167 and 61802348) and the Natural Science Foundation of Zhejiang Province (LY20F030017).

[1]Authors are with College of Computer Science and Technology, Zhejiang University of Technology, 310023 Hangzhou, China. zjh@ieee.org

[2]Shenyong Chen is with School of Computer Science and Engineering, Tianjin University of Technology. sy@ieee.org

[3]Honghai Liu is with School of Computing, University of Portsmouth, Portsmouth, UK. honghai.liu@ieee.org

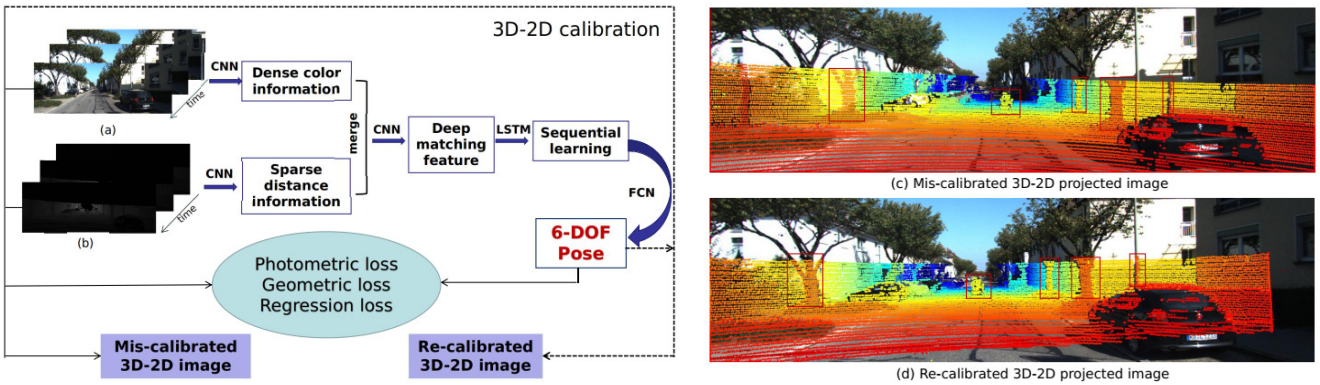


Fig. 1. Architecture of CalibRCNN. (a) the RGB images of consecutive frames; (b) depth maps of consecutive frames generated from the mis-calibrated point clouds. By projecting 3D point cloud to the corresponding 2D RGB image we can obtain images similar to (c) and (d), where the color of the projected point represents its depth value. (c) is a 3D-2D projection image converted by mis-calibration parameters, while (d) is the projection image after calibration using our network. Red rectangles show the difference before and after calibration.

to extract the different kinds of features from the single-frame depth map and the RGB image, respectively.

It is worth noting that end-to-end deep learning has made outstanding achievements in 3D target detection [17], depth estimation [18], pose estimation [4], etc., which have reference meaning for multi-sensor extrinsic calibration. Schneider et al. [15] propose the RegNet, the first deep convolutional neural network (CNN) for LiDAR-camera calibration. After that, the CalibNet [16] is proposed to solve the calibration problem by reducing the dense photometric error and the dense point cloud distance error between the mis-calibrated depth map and the target depth map, and consequently it increases the generalization ability of the model.

We find that most of the existing end-to-end calibration methods do not take into account the pose transformation among successive frames. But the geometric relationship constrained by relative pose transformation does exist among successive pairs of LiDAR and camera data. As demonstrated by many traditional online multi-sensor calibration methods, extrinsic parameters will converge along with inputting successive frames.

We can borrow ideas from the end-to-end deep learning based odometry frameworks, which take the relative pose transformation among sequential frames into account, because the calibration between LiDAR and camera is equivalent to estimate the relative pose between the LiDAR and the camera. DeepVo [3] is a successful method towards end-to-end visual odometry. It takes into account the importance of sequential dependence and complex motion dynamics of an image sequence. [4] and [5] are also methods for camera relative pose estimation. They mainly use synthetic view constraints and epipolar geometry constraints between successive frames of RGB images for model optimization. Inspired by these methods, our system is designed to optimize the calibration using the constraint relationship between successive frames and does improve the calibration accuracy.

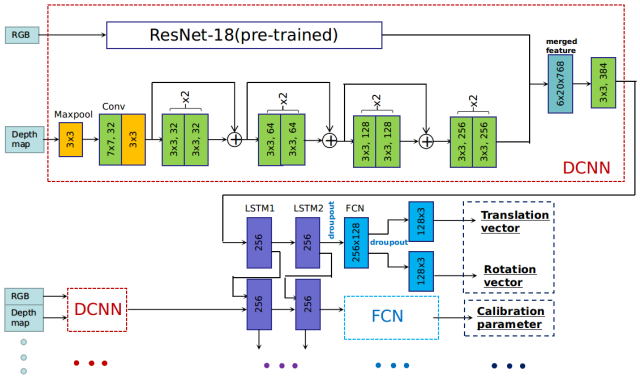


Fig. 2. Architecture of the proposed Calibration Recurrent Convolutional Neural Network.

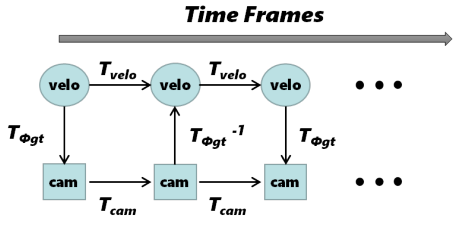


Fig. 3. Pose transformation relationship diagram of continuous frames of LiDAR and camera data.

III. Method

In our method, we first assume an initial estimation of the transformation parameters, T_{init}, which is not accurate enough and causes a mis-calibrated depth map to some extent. Then, our model can predict a certain range of deviations of the transformation parameter, $T_{decalib}$, by which the external calibration parameter T_ϕ can be calibrated as accurate as possible.

A. Data preprocessing

During training phase, the required inputs are three consecutive RGB images taken by the camera, three corresponding point clouds collected by the LiDAR, and the basic parameters required for the calculation of the loss function, such as the camera intrinsic parameters *K*, the camera pose between two frames, and the ground truth transformation parameters T_ϕ between LiDAR and camera. For the original point cloud data, the transformation parameters T_{init} = T_ϕ, projecting a 3D point $[x, y, z]$ to a pixel $[u, v, z_c]$ in the image coordinate system is consistent in each frame. The projection formula is shown in Eq.(1), where z_c is the pixel value.

$$\begin{bmatrix} u * z_c \\ v * z_c \\ z_c \\ 1 \end{bmatrix} = KT_{init} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1}$$

In order to ensure the diversity of the training data, we use the transformation parameters $T_{decalib}$ to mis-calibrate. Randomly generated transformation $T_{decalib}$ can produce a lot of training data, as it is also used in [16]. Each input consists of three adjacent pairs of depth maps and images, where three depth maps are projected by the same $T_{decalib}$. According to Eq.(2), the calibration transformation expected from the network output is $T_{calib} = T_{decalib}^{-1}$.

$$T_{init} = T_{decalib} T_\phi \tag{2}$$

It is known that there is a certain consistent constraint between the camera and the LiDAR among successive frames, as shown in Fig.3. We can obtain the coordinate transformation parameters T_{velo} between the two frames of point cloud by the inter-camera pose transformation T_{cam} and the ground truth LiDAR-Camera transformation parameters $T_{\phi_{gt}}$, according to Eq.(3).

$$T_{velo} = T_{\phi_{gt}} T_{cam} {T_{\phi_{gt}}}^{-1} \tag{3}$$

B. Network framework

The network is mainly composed of two branches, like [15], [16], which respectively extract features from 2D images and depth maps (see Fig. 2). For the RGB branch, we use the convolutional layer of the pre-trained ResNet-18 network [19]. For the depth map branch, we use a network similar to the ResNet-18 structure. But we halve the number of filters, and find it is enough to extract features contained in the mis-calibrated depth map. Then, we fuse the features output from the two network branches, and perform a global feature aggregation through a convolution layer and a LayerNorm layer. Next, the obtained global feature pair of each frame is used as an input to the LSTM layer, extracting temporal information between consecutive frames for sequential learning [3]. Finally, we use the fully connected layer to parse the translation and rotation information from the deep features output by the LSTM, and predict the translation vector τ and the rotation vector γ. Here the rotation vector γ can be converted to a rotation matrix $R \in SO(3)$ by the well-known Rodrigues formula. Combining with translation vector $\tau \in R^3$ gives us a 3D transformation matrix $T_{calib} \in SE(3)$, which is expected to be the inverse of $T_{decalib}$, and defined as

$$T_{calib} = \begin{bmatrix} R & \tau \\ 0 & 1 \end{bmatrix} \tag{4}$$

C. Loss function

Depth map calibration error. The input depth map $D_{miscalib}$ can be temporarily calibrated using the transformation matrix T_{pre} output by the network, generating the predicted depth map D_{pre}, as shown in Eq.(5).

$$\begin{bmatrix} u_p * z_{cp} \\ v_p * z_{zp} \\ z_{cp} \\ 1 \end{bmatrix} = KT_{pre}K^{-1} \begin{bmatrix} u * z_c \\ v * z_c \\ z_c \\ 1 \end{bmatrix} \tag{5}$$

Note that here the T_{pre} will be continuously refined along with the model training, and finally becomes the accurate transformation matrix T_{calib}.

At the same time, using the ground truth transformation matrix $T_{decalib}^{-1}$ to calibrate the input depth map that is mis-calibrated by $T_{decalib}$, we can obtain the ground truth depth map D_{gt}. In order to optimize the calibration capability of the network, it is necessary to quantify the difference between two depth maps as a loss function. For each projection point $p(u, v, z_c)$ in the $D_{miscalib}$, there is a corresponding point $p_{pre}(u_p, v_p, z_{cp})$ and $p_{gt}(u_g, v_g, z_{cg})$ in D_{pre} and D_{gt}, respectively. We define the depth map difference loss function as

$$L_D = \frac{\sum_{p \in D}^{N} \left\| p_{pre} - p_{gt} \right\|_2}{N} \tag{6}$$

where N is amount of points used in this loss.

Synthetic view constraint. Give the initial transformation parameter, T_{init}, we can substitute the predicted calibration parameters T_{pre} into the right-hand side of Eq.(2) to compute the predicted T_ϕ, which can be further substituted into Eq.(3) to compute the transformation T_{cam} between two camera poses by using the obtained T_{velo}. Meanwhile, we can use T_{pre} to obtain a calibrated depth map. From the depth map and the camera transformation, a synthesis view can be obtained for RGB images of consecutive frames. With reference to the loss function of the depth estimation and the camera pose estimation in [4], we establish geometric constraints between successive 2D frames. It should be noted that the synthesis view of continuous 2D frames of the monocular camera has many restrictions and is not fully applicable to the current 2D image. Therefore, we propose to leverage matched SIFT feature points [20] as reference points for the loss calculation. Refer to the view synthesis method in [21], assuming p_1 denotes the selected SIFT point in the target image I_1, its projection p_2 in the source image I_2 is represented by

$$p_2 \sim KT_{cam} D_1(p_1) K^{-1} p_1 \tag{7}$$

Then, we can obtain a synthesis image I_2' using source frames I_2 by bilinear sampling. For each point p_2 in I_2', its value $I_2'(p_2)$ is interpolated by the neighbor points of p_2 in the source image I_2. The synthetic view constraint photometric loss can be formulated as

$$L_S = \frac{\sum_{p \in I_2'}^{N} |I_1(p) - I_2'(p)|}{N} \tag{8}$$

Epipolar geometry constraint. Inspired by [5], it is possible to optimize the transformation T_{cam} between two camera poses by the epipolar geometry constraint. As shown in Eq.(3), the optimization of T_{cam} can refine the precision of the LiDAR-camera calibration parameter T_ϕ. Thus, the epipolar geometry constraint [22] can also be used to optimize LiDAR-camera calibration prediction. Supposing p_i and q_i are pairs of matched SIFT feature points in the two adjacent 2D image frames, the epipolar geometry constraint can be formulated as

$$q_i K^{-T} E K^{-1} p_i = 0 \tag{9}$$

where, E is the essential matrix related to the pose between two frames. Thus, we can calculate the epipolar geometry loss L_{geo} as follows:

$$L_{geo} = q_i K^{-T} E K^{-1} p_i \tag{10}$$

Global regression error of calibration parameters. In order to achieve the better accuracy of the calibration parameters from our network regression, we also compute the Euclidean loss function between the predicted parameters and the ground truth parameter,

$$\begin{aligned} L_P &= L_{translation} + L_{rotation} \\ &= \|\tau_{pre} - \tau_{gt}\|_2 + \|\gamma_{pre} - \gamma_{gt}\|_2 \end{aligned} \tag{11}$$

Our final loss function consists of a weighted sum of above losses:

$$L_{final} = \lambda_1 L_D + \lambda_2 L_S + \lambda_3 L_{geo} + +\lambda_4 L_P \tag{12}$$

where the weight λ_i of each loss. After experimental analysis, we found that the above four loss functions all have an indispensable effect on the calibration accuracy of the model, finally, we set the weights to 0.01, 0.1, 0.01, and 1, respectively.

IV. Experiment

In order to verify the proposed method[1], the KITTI-odometry dataset is selected to train and test the model. The camera and LiDAR datas used in the experiment are synchronized. In this section, the specific settings in the experiment, the training process of the model, and the qualitative and quantitative analysis of the experimental results will be explained in detail.

[1]The source code of our implementation can be found at https://github.com/zjut-jianhuazhang/CalibRCNN

A. Dataset Preparation

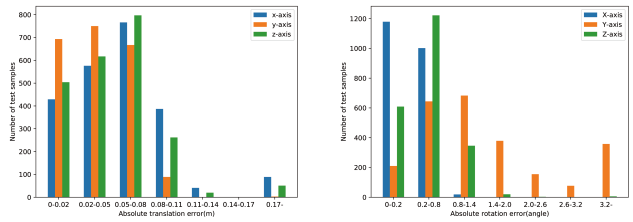
First, we use three consecutive pairs of camera images and LiDAR point clouds as an input sample, and apply random transformation parameters $T_{miscalib}$ to mis-calibrate each set of point cloud data. The deviation range of mis-calibration is set to $\pm 10°$ rotation and $\pm$ 0.25m translation of any axis. By projecting the point cloud into the image plane by the mis-calibration parameters and camera intrinsic parameters, the mis-calibrated sparse depth map can be obtained. The procedure of mis-calibration is similar to [16]. The coordinate transformation parameter T_{velo} between the two point clouds are calculated according to Eq.(3). After preprocessing the 00-06 sequences of the KITTI-odometry dataset, we take 90% frames from each sequence as the training set, and the remaining 10% as the test set. In addition, we also use parts of Kitti-raw 0926 sequence as the test data which have unfamiliar scenes and different intrinsic parameters.

B. Training Details

The training of the network is performed with the Adam Optimizer [23], using an initial learning rate 1e-4. We decrease the learning rate by a factor 0.5 every few epochs. Besides, in our work, the deep RNN is designed by stacking two LSTM layers with the hidden states of a LSTM being the input to the other one, as show in Fig. 2, and the hidden states of each LSTM layers are set to 256. In order to prevent overfitting, we apply the regularization loss and set the regularization parameter to 0.001.

C. Results

We evaluate the proposed model by setting different initial mis-calibrated ranges for training and test data, and verify its accuracy of predicting calibration parameters. By preliminary experiments, we find that most of the results have a satisfactory calibration accuracy, but few calibration results still exhibit considerable deviations. Fig.4 illustrates the calibration error distribution of the whole experimental results over a wide error range of initial mis-calibration. In the following parts we will explain our experiments from several aspects in detail.



(a) (b)

Fig. 4. Calibration error distribution of the whole test dataset over a wide range of initial mis-calibration. (a) shows the translation error distribution of test results. (b) shows the rotation error distribution of test results.

Calibration examination on the overall test dataset. Our system performs exceptionally well in translation parameters

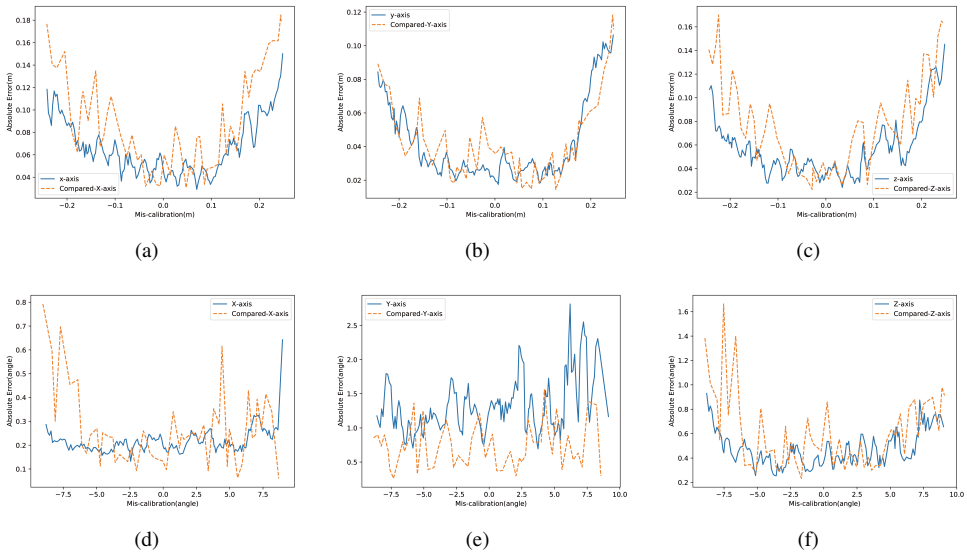


Fig. 5. Calibration error of the model for different initial deviation calibrations on each axis. The solid lines in the figure are the test results on the entire dataset, while the dotted lines in the figure is the results of testing in the environmental dataset with a wide field of view.

estimation. The mean absolute error (MAE) value for translation on the test set is (X: 0.066m, Y:0.042m, Z: 0.055m). As for rotation estimation, through preliminary test, we find that the Y-axis is the most difficult to calibrate among the three axes of rotation. This is because the rotation on the Y-axis has a greater influence on the projection depth variation of 3D point clouds on image plane. The mean absolute error (MAE) value for rotation angles on the whole test set is (X: $0.216°$, Y: $1.33°$, Z: $0.478°$). The solid lines in Fig.5 further illustrate the absolute errors, against a widespread variation in mis-calibrations.

By analyzing the test results, we find that the model performed better in an environment with a straight road structure, than in the complex environment with occlusion ahead, especially in terms of rotational calibration. This is because a wide field of view is able to obtain more abundant deoth information, and the projection of the long-distance 3D point cloud on the 2D image is more sensitive to the rotation parameters, especially the rotation on Y-axis. To this end, we eliminate data pairs from relatively complex structural environments in the test data set and perform calibration tests on the remaining data. We show in Fig.5 the calibration errors of the model for the six axes of rotation and translation under different initial mis-calibrations. Through this experiment, it can be found that the calibration result of the rotating Y-axis has been greatly improved, as show in Fig.5(e). It no longer has a large calibration deviation, and the average absolute error of this axis is about $0.64°$.

Calibration on other dataset. We evaluate the proposed model with the Kitti-raw 0926 driving dataset. The experimental results show that our system has good generalization ability. It can also get well calibration in untrained and unfamiliar datasets. This proves that, to a certain extent, our system does not rely on constant parameters of the sensor such as camera intrinsic. The calibrated translation and rotation deviations on the whole sequence are (X: 0.102m, Y:0.044m, Z: 0.081m) and (X: $0.408°$, Y: $3.59°$, Z: $0.533°$), respectively, while the calibration deviations on the scenes with wider view are (X: 0.078m, Y:0.032m, Z: 0.062m) and (X: $0.21°$, Y: $2.21°$, Z: $0.50°$). It can be proved again that in the wide view scene, our model works better.

Compared with other CNN based methods. There are few existing calibration methods based on neural networks. We take RegNet [15] and CalibNet [16] as examples to compare. Their experimental datasets and evaluating process are different, so it is difficult to achieve complete quantitative comparison. For RegNet [15], the calibration results obtained by its single model are not clearly stated in the paper, and we can only know from the graph that the calibration error of each single model, which is trained with different calibration deviation range, is greater than the calibration error of our single model. It is of great reference value that it applies different models for iterative calibration, and achieves a mean error of (X: 0.07m, Y: 0.07m, Z: 0.04m) in translation and (X: $0.24°$, Y: $0.25°$, Z: $0.36°$) in rotation for a larger initial deviation.

CalibNet [16] strives to optimize the calibration model from the underlying geometric problems, thereby improving the generalization ability of the model. The advantage is that it can effectively improve the rotation error, while the disadvantage is that the translation and rotation cannot be correctly estimated in single iteration. Its rotation error is (X: $0.15°$, Y: $0.9°$, Z: $0.18°$). Then it trains the model for predicting the translation parameters separately, and obtain a translation error of (X: 0.12m, Y: 0.03m, Z: 0.08m) with given CalibNet rotation estimates.

Our model is only trained as a single model for translation and rotation calibration, and the calibration results can be comparable to the multi-model iterative calibration of other methods. The comparison results are listed in Table 1. Calib-

TABLE I

CALIBRATION RESULTS OBTAINED BY DIFFERENT METHODS.

Calib. err.	Translation(m)				Rotation(angle)			
	X	Y	Z	τ	X	Y	Z	γ
RegNet	0.07	0.07	0.04		0.24	0.25	0.36	
CalibNet	0.042	0.016	0.072					
	0.12	**0.035**	0.079	0.149	**0.15**	0.9	**0.18**	0.930
Ours	0.066	0.044	0.055	0.097	0.216	1.33	0.478	1.423
	0.062	0.043	**0.054**	**0.093**	0.199	**0.64**	0.446	**0.805**

Net uses two models for rotation and translation calibration separately. And the translation calibration is performed by two approaches: 'Given ground truth rotation parameters (in the 2nd row)' and 'Given CalibNet rotation estimations (in the 3rd row)'. In the 4th row, we list our results on the whole test dataset. And in the 5th row, the results are obtained from the wide view scenes, and consequently have better accuracy. The best results are indicated by bold font in the last three rows, where the comparison is related fair. The columns of τ and γ are the root square error of all axis in terms of translation and rotation, respectively. It is worth noting that our results is better than to that of CalibNet in the wide view scenes, even the CalibNet is trained through two models.

V. CONCLUSION

In this paper, we introduce a novel method for extrinsic calibration between 3D LiDAR and 2D camera based on a deep neural network by combining CNN and LSTM. Compared to most existing methods, our method does not need to any human intervention, and enables online real-time calibration, which infers the 6-DOF rigid body transformation. We only need to train one model on wide range of initial error calibration. Because the optimization of the model is based on the underlying geometric problem of 3D-2D calibration, it has good generalization ability, adapting to unfamiliar scenes and different intrinsic parameters. In the test dataset, our model yields a calibration error of 9.3 cm for translation and 0.805° for rotation, which is better than the state-of-the-art methods. In the future, we will work to improve the generalization ability of the model on the basis of ensuring the calibration accuracy.

REFERENCES

[1] Z. Wang, J. Zhang, S. Chen, C. Yuan, J.Zhang, and J. Zhang. Robust high accuracy visual-inertial-laser slam system. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[2] J. Zhang and S. Singh. Laser-visual-inertial odometry and mapping with high robustness and low drift. *Journal of Field Robotics*, 35(5):1242–1264, 2018.

[3] S. Wang, R. Clark, H. Wen, and N. Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2050, May 2017.

[4] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6612–6619, July 2017.

[5] T. Shen, Z. Luo, L. Zhou, H. Deng, R. Zhang, T. Fang, and L. Quan. Beyond photometric loss for self-supervised ego-motion estimation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6359–6365, May 2019.

[6] L. Zhou, Z. Li, and M. Kaess. Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[7] Zachary Jeremy Taylor. Automatic markerless calibration of multi-modal sensor arrays. 2015.

[8] Z. Pusztai and L. Hajder. Accurate calibration of lidar-camera systems using ordinary boxes. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017.

[9] J. Domhof, J. F. P. Kooij, and D. M. Gavrila. An extrinsic calibration tool for radar, camera and lidar. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8107–8113, May 2019.

[10] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster. Automatic camera and range sensor calibration using a single shot. In *2012 IEEE International Conference on Robotics and Automation*, pages 3936–3943, May 2012.

[11] J. Jiang, P. Xue, S. Chen, Z. Liu, X. Zhang, and N. Zheng. Line feature based extrinsic calibration of lidar and camera. In *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pages 1–6, Sep. 2018.

[12] J. Castorena, U. S. Kamilov, and P. T. Boufounos. Autocalibration of lidar and optical cameras via edge alignment. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[13] Hsiang-Jen Chien, R. Klette, N. Schneider, and U. Franke. Visual odometry driven online calibration for monocular lidar-camera systems. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2848–2853, Dec 2016.

[14] H. Liu, Y. Liu, X. Gu, Y. Wu, F. Qu, and L. Huang. A deep-learning based multi-modality sensor calibration method for usv. In *2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM)*, pages 1–5, Sep. 2018.

[15] N. Schneider, F. Piewak, C. Stiller, and U. Franke. Regnet: Multimodal sensor registration using deep neural networks. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1803–1810, June 2017.

[16] G. Iyer, R. K. Ram, J. K. Murthy, and K. M. Krishna. Calibnet: Geometrically supervised extrinsic calibration using 3d spatial transformer networks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1110–1117, Oct 2018.

[17] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1355–1361, May 2017.

[18] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 239–248, Oct 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[20] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[21] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. Efros. View synthesis by appearance flow. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 286–301, Cham, 2016. Springer International Publishing.

[22] J. Zhang, C. Wang, and W. Liao. An epipolar geometry constraint based view synthesis algorithm. In *2009 2nd International Congress on Image and Signal Processing*, pages 1–5, Oct 2009.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.